REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-18-2006	Final Report	5/25/02 – 12/30/05

4. TITLE AND SUBTITLE
Time Domain Non-Linear SAR Processing

5a. CONTRACT NUMBER

5b. GRANT NUMBER
N00014-02-1-0746

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Mehrdad Soumekh

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Department of Electrical Engineering
332 Bonner Hall
State University of New York
Buffalo, NY 14260

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Ballston Center Tower One
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The main objectives of this effort were to develop a theoretical foundation for Time Domain Non-Linear SAR Processing, and corresponding DSP algorithms to efficiently implement the process on existing computer architectures. We formulated the equations to convert a flight path GPS/INS data into ECEF data that were suitable for mapping into a desired slant imaging plane. The GPS data of an existing SAR platform were used for testing this mapping. Codes were developed for simulating the non-linear SAR signature of targets for a given set of flight path GPS data. We established the mathematical foundation and MATLAB codes for backprojection and wavefront image formation algorithms for on a non-linear SAR trajectory using multi-processor computers. We conducted parallel computing for the proposed reconstruction algorithms on a shared memory SGI computer and a distributed memory Dell computer using MatlabMPI and C. We also converted the algorithms into parallel Matlab code and created graphical user interfaces for both programs. Parallel algorithms for a SAR-MTI problem were also developed. The two imaging algorithms were studied and tested using both actual and simulated SAR data. These algorithms have also been chosen and implemented for a US Army platform under the WAAMD (Wide Area Airborne Mine Detection) Program.

15. SUBJECT TERMS
Synthetic aperture radar, non-linear flight path

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mehrdad Soumekh
Unclassified	Unclassified	Unclassified			19b. TELEPHONE NUMBER *(include area code)* (716) 645-3115, x2138

Time Domain Non-Linear SAR Processing

Final Report

Supported by the Office of Naval Research
Award # N00014-02-1-0746
Program Directors: Dr. M. Pollock and Mr. G. McNeal
Principal Investigator: Mehrdad Soumekh

January 18, 2006

Contents

1 Abstract

The main objectives of this effort were to develop a theoretical foundation for Time Domain Non-Linear SAR Processing, and corresponding DSP algorithms to efficiently implement the process on existing computer architectures. We formulated the equations to convert a flight path GPS/INS data into ECEF data that were suitable for mapping into a desired slant imaging plane. The GPS data of an existing SAR platform were used for testing this mapping. Codes were developed for simulating the non-linear SAR signature of targets for a given set of flight path GPS data. We established the mathematical foundation and MATLAB codes for backprojection and wavefront image formation algorithms for on a non-linear SAR trajectory using multi-processor computers. We conducted parallel computing for the proposed reconstruction algorithms on a shared memory SGI computer and a distributed memory Dell computer using MatlabMPI and C. We also converted the algorithms into parallel Matlab code and created graphical user interfaces for both programs. Parallel algorithms for a SAR-MTI problem were also developed. The two imaging algorithms were studied and tested using both actual and simulated SAR data. These algorithms have also been chosen and implemented for a US Army platform under the WAAMD (Wide Area Airborne Mine Detection) Program.

2 Non-linear SAR Signal Processing and Imaging

The geometry for a non-linear SAR system is shown in Figure 1. The main task to be performed under this project is to provide a SAR image in a fixed (desired) target coordinate system for the human (Navy) operator irrespective of the portion of a general non-linear flight path (slow-time interval) that is used for coherent integration. This reduces the complexity of the information that is conveyed to the human operator by the displayed SAR image. Any other information constructed by the computer from the raw SAR phase history data or the complex formed image (for example, change detection, MTI, GMTI, etc.) can be displayed in the same coordinate system; this facilitates the integration of these data with the visual SAR image by the operator.

The basic principle (starting point) behind this approach is the general *squint* SAR image formation. The frames of the SAR video (time series imagery) are generated in a common spatial coordinate system though they are originated from different aspect angles or, equivalently, Doppler data. Each frame is formed via the (slow-time) coherent integration of a contiguous segment of the flight path that, in general, is *non-linear*; the segments may or may not overlap.

We have studied two SAR imaging methods for this purpose: time domain correlation-based *backprojection* algorithm, and Gabor's *wavefront* reconstruction method. These algorithms are guided by the GPS/INS data and, if available, DTED (or any other survey data). The main objective is to develop accurate and computationally-efficient *recursive* versions of these two methods that provide the operator with a video SAR image as the SAR phase history data are being collected in the slow-time domain. These methods are discussed in the next sections.

3 SAR Reconstruction via Time Domain Correlation and Backprojection

SAR imaging may be formulated via convolving the SAR signal with a *shift-varying* filter [bar]. In this section, we present two correlation-based SAR digital reconstruction algorithms which utilize this principle, and their application in non-linear SAR.

3.1 Time Domain Correlation Algorithm for Linear SAR

The basic principle behind the TDC imaging method is simply *correlation* implementation of the SAR matched filtering [bar]. Denote the transmitted radar signal with $p(t)$ where t is the fast-time domain. Suppose we are interested in forming the target function at a set of two-dimensional sampled points (x_i, y_j)'s in the spatial domain. The TDC processor correlates the SAR signature at a given grid point (x_i, y_j) which is

$$p[t - \frac{2\sqrt{x_i^2 + (y_j - u)^2}}{c}], \tag{1}$$

with the measured SAR data in the fast-time and slow-time (t, u) domain, that is, $\mathbf{s}(t, u)$. The result is a measure of reflectivity at that grid point. The imaging equation for TDC is

$$\begin{aligned} f(x_i, y_j) &= \int_u \int_t \mathbf{s}(t, u)\, p^*\Big[t - \frac{2\sqrt{x_i^2 + (y_j - u)^2}}{c}\Big]\, dt\, du \\ &= \int_u \int_t \mathbf{s}(t, u)\, p^*[t - t_{ij}(u)] dt\, du, \end{aligned} \tag{2}$$

where $p^*(t)$ is the complex conjugate of $p(t)$, and

$$t_{ij}(u) = \frac{2\sqrt{x_i^2 + (y_j - u)^2}}{c}, \tag{3}$$

is the round trip delay of the echoed signal from the target located at $(x, y) = (x_i, y_j)$. This is depicted in Figure 1 for a general non-linear slow-time u domain.

In practice, the two-dimensional integral in the fast-time and slow-time domain (t, u) is converted into a double sum over the available discrete values of (t, u), that is, the domain of the measured SAR data. The reconstruction is performed for discrete values of the spatial (x, y) domain on a uniform grid, that is, (x_i, y_j)'s. To reduce the numerical errors, the measured signal $\mathbf{s}(t, u)$ is *upsampled* (interpolated) in both the fast-time and slow-time domains.

The correlation for the TDC method may also be performed in the (ω, u) domain via utilizing the following identity which can be obtained from the general Parseval's theorem:

$$\int_t \mathbf{s}(t, u)\, p^*[t - t_{ij}(u)]\, dt = \int_\omega s(\omega, u)\, P^*(\omega)\, \exp[j\omega t_{ij}(u)]\, d\omega. \tag{4}$$

Substituting this identity in the TDC imaging equation, one obtains the following:

$$f(x_i, y_j) = \int_u \int_\omega s(\omega, u)\, P^*(\omega)\, \exp[j\omega t_{ij}(u)]\, d\omega\, du. \tag{5}$$

It is not difficult to see that the (ω, u) domain correlation method to reconstruct the target function can be converted into the convolution-based reference signal matched-filtering of the range stack reconstruction method.

3.2 Backprojection Algorithm for Linear SAR

We denote the fast-time matched-filtered SAR signal via

$$\mathbf{s}_M(t,u) = \mathbf{s}(t,u) * p^*(-t), \tag{6}$$

where $*$ denotes convolution in the fast-time domain. Using this in the TDC equation, we obtain

$$\begin{aligned} f(x_i,y_j) &= \int_u s_M\Big[\frac{2\sqrt{x_i^2+(y_j-u)^2}}{c}, u\Big]\,du \\ &= \int_u s_M[t_{ij}(u),u]\,du \end{aligned} \tag{7}$$

where

$$t_{ij}(u) = \frac{2\sqrt{x_i^2+(y_j-u)^2}}{c} \tag{8}$$

is the round trip delay of the echoed signal for the target at (x_i,y_j) when the radar is at $(0,u)$. Thus, to form the target function at a given grid point (x_i,y_j) in the spatial domain, one could coherently add the data at the fast-time bins that corresponds to the location of that point for all synthetic aperture locations u.

This algorithm is known as the *backprojection* SAR reconstruction. This is due to the fact that for a given synthetic aperture location u, the fast-time data of $s_M(t,u)$ are traced back in the fast-time domain (backprojected) to isolate the return of the reflector at (x_i,y_j). A block diagram for the backprojection algorithm is shown in Figure 2.

To implement this method in practice, the available discrete fast-time samples of $s_M(t,u)$ must be interpolated to recover

$$s_M[t_{ij}(u),u].$$

If a sufficiently accurate interpolator was not used, this would result in the loss of high-resolution information.

3.3 Backprojection Algorithm for Non-Linear SAR

As we mentioned earlier, the backprojection reconstruction algorithm is based on tracing back the signature of a given reflector in the fast-time domain of the matched-filtered signal $s_M(t,u)$ at a given slow-time u, and coherently adding the results at the available u values. The algorithm can be easily modified to incorporate a known non-linear motion for the flight path of the radar-carrying aircraft. This is shown in the following.

At the slow-time u, the radar is located at the coordinates $[x_r(u),y_r(u)]$; see Figure 1. For a given reflector at the spatial coordinates (x_i,y_j), the signature of this target in the matched-filtered signal can be traced back to the fast-time:

$$t_{ij}(u) = \frac{2\sqrt{[x_i-x_r(u)]^2+[y_j-y_r(u)]^2}}{c}. \tag{9}$$

Thus, the backprojection reconstruction becomes

$$f(x_i,y_j) = \int_u \mathbf{s}_M\Big[\frac{2\sqrt{[x_i-x_r(u)]^2+[y_j-y_r(u)]^2}}{c}, u\Big]\,du. \tag{10}$$

The algorithm can still be identified using the block diagram in Figure 2 with $t_{ij}(u)$ defined via the above equation that depends on the non-linear flight-path.

3.4 Recursive Video Image Formation

To formulate image formation for the generation of a video SAR, we denote the discrete slow-time points at which the target area is irradiated with the radar signal by u_n, $n = 1, 2, 3, 4, ...$ We also denote the backprojection image that is formed with the data from a *single* slow-time (transmission) at u_n via

$$\Delta f_n(x_i, y_j) = \mathbf{s}_M\Big[\frac{2\sqrt{[x_i - x_r(u_n)]^2 + [y_j - y_r(u_n)]^2}}{c}, u_n\Big]. \tag{11}$$

We identify the above reconstruction as the n-th *differential* SAR image. Note that this image has an extremely poor azimuth resolution since it is formed using a single PRI.

Suppose the user wishes to integrate K PRIs to form a single frame of the video SAR image. Thus, at the N-th slow-time PRI, the image formed can be expressed as:

$$f_N(x_i, y_j) = \sum_{n=N-K+1}^{N} \Delta f_n(x_i, y_j). \tag{12}$$

Note that the image that is constructed at the previous frame, that is, $(N-1)$-st is

$$f_{N-1}(x_i, y_j) = \sum_{n=N-K}^{N-1} \Delta f_n(x_i, y_j). \tag{13}$$

Thus, we can use the following *recursive* or *updating* equation to form the N-th frame using the previous frame and two differential SAR images:

$$f_N(x_i, y_j) = f_{N-1}(x_i, y_j) + \Delta f_N(x_i, y_j) - \Delta f_{N-K}(x_i, y_j). \tag{14}$$

The main issue in implementing the above-mentioned methods is the computational cost particularly if the user wishes to realize these algorithms in (near) real-time. Recently a method known as quad-tree backprojection has been introduced to reduce the computational load of the SAR backprojection algorithm; this method uses certain approximations. As we mentioned earlier, in the case of the original backprojection method, one has to use an accurate interpolator in the fast-time domain. For example, the Matlab code in [s99] uses a 1:100 upsampling FFT interpolator; anything less accurate, e.g., a cubic spline or bilinear interpolation, would result in noticeably less accurate results. In the case of the quad-tree algorithm, the problem is further compounded due to the need for decimation in the slow-time domain at each step of the quad-tree.

A subtle way that an approximation contaminates a SAR image is the manner it distorts the phase data without altering magnitude information; i.e., the ideal image and a quad-tree image may look the same though they carry different phase information. (Phase information turns out to be an important feature in SAR.) Based on our processing of realistic SAR data, the quad-tree is not a viable option for SAR imaging. We believe the best option for implementing the backprojection method is to preserve the integrity of the coherent data. The main challenge that we face for the backprojection-based video non-linear SAR image formation is to develop efficient and accurate algorithms to implement the above-mentioned recursive backprojection method.

3.5 Backprojection Program and Parallelization

The original Matlab program is bp_sim_patch.m. It was parallelized using MatlabMPI and a GUI was created for it by using *guide*. The parallel version is named bp.m and GUI files start with bp_function.m. Figure 3 shows the backprojection output image.

There are two *for* loops in bp_sim_patch.m for which parallelization is required. The structure of those loops in the sequential program is as follows:

```
for (ii = 1: mx)
    for (jj = 1: my)
        ... (Computation algorithm)
    end
end
```

Before creating the parallel version, these two *for* loops were converted into a single *for* loop. The new structure is as follows:

```
for (ind = 1: mx*my)
    ii = ceil (ind/my);
    jj = ind-(ii-1)*my;
        ... (Computation algorithm)
end
```

The modified loop, called *ind loop*, performs the same operations as the original, but it is conceptually easier to understand and partition for parallel processing. This loop is in turn nested in another *for* loop called the *j loop*. For each iteration of the *j loop*, the *ind loop* has *mx*my* number of iterations. To parallelize the *ind loop*, the *mx*my* number of iterations is split among the number of processors available to the user. The new structure is as follows:

```
parts = ceil(mx*my/procs);
start = rank*parts+1;
ending = (rank+1)*parts;
for (ind = start:ending)
    ii = ceil (ind/my);
    jj = ind-(ii-1)*my;
        ... (Computation algorithm)
end
```

For example, if the user specified four processors are available for parallel processing, then each processor will perform *mx*my/4* number of iterations of the *ind loop*. The *ind loop* is nested inside the *j loop* and therefore is repeated *j* number of times. But for each repetition of the *ind loop*, *ind=mx*my* is constant. The parallelization is designed so that for each repetition of the *ind loop*, each processor processes the same indexes.

In other words, for the "j=1" loop, session 1 will process, say, ind=1:100, and for the "j=2" loop, session 1 will also process ind=1:100, and so on. When the *j loop* has completely finished running, the slave sessions will send their computed data to the master session. The master session combines these segmented data and outputs the result.

Simulation Results

The simulation results were obtained on both machines Stingray and Hellfire. Here, Stingray is an SGI Origin 2000 machine with 16 MIPS 300 MHz IP27 R12000 processors. The share memory size is 8192MB and the OS used is IRIX64 version 6.5. Hellfire is a DELL cluster with 33 computenodes and there are two Intel Xeon 2.4 GHz processors per computenode. The OS used is Linux 2.4.18-4smp (i686-linux). The MATLAB used on both machines are version 6.5.

Table 1 shows the runtime of the parallelized backprojection program. Every set of parameters is only run once because of the length of time it needs to run. This table shows that the speedup of this program is not very significant when the number of the nodes goes up. This is due to the effect of the diminishing returns stated in the Amdahl's law.

The most common measure of parallel programs' performance is *speedup*. For number

Table 1: Performance Comparison of the Backprojection Program

Number of nodes	1	2	4	4	4	8	16	32
range(m)	400	400	400	400	200	400	400	400
azimuth(m)	600	600	600	300	600	600	600	600
sub-range(m)	50	50	50	50	50	50	50	50
sub-azimuth(m)	50	50	50	50	50	50	50	50
Runtime(sec) on Stingray	4747.8	3083.0	2239.7	1774.5	1821.4	1818.9	1665.7	N/A
Runtime(sec) on Hellfire	1539.2	1212.0	1025.9	960.2	969.1	953.2	917.2	921.7
Ratio (Hellfire/Stingray)	32.4%	39.3%	45.8%	54.1%	53.2%	52.4%	55.1%	N/A

of nodes N, speedup is the ratio between between the serial and parallel runtime.

$$speedup(N) = \frac{time_{serial}}{time_{parallel}(N)} \tag{15}$$

Amdahl's law states that if f is the fraction of a program which is sequential (i.e. cannot benefit from parallelization), and 1-f is the fraction that can be parallelized, then the speedup that can be achieved by using N nodes satisfies

$$speedup(N) \leq \frac{1}{f + \frac{1-f}{N}}. \tag{16}$$

Figure 4 shows the theoretical maximum speedups for parallel programs with different values of f. As we can see, how much of a program can be parallelized is an important factor in determining the maximum possible speedup. Table 2 shows the estimated value of f for the three programs in this report. It should be noted that the value of f for the same program may be different on different machines. The value of f in this table is then used to calculate the theoretical boundary, which is shown and compared with the actual speedups in the following figure.

Figure 5 shows the speedup as a function of the number of the nodes for the backpro-

Table 2: The value of f for different programs and machines

	Stingray	Hellfire
Backprojection	0.302	0.561
Wavefront Reconstruction	$< 10^{-4}$	$< 10^{-4}$
Multi-Channel SAR-MTI	$< 10^{-5}$	$< 10^{-5}$

jection program. We can see the actual speedups reach the theoretical boundary when the number of nodes is small, but there is almost no improvement when it reaches a certain point. According to Amdahl's law:

$$\lim_{N\to\infty} speedup(N) \leq \frac{1}{f}. \tag{17}$$

Therefore for a program with large f, the improvement of performance is mainly obtained for a few nodes.

4 SAR Wavefront-Based Signal Processing and Reconstruction

4.1 Wavefront Reconstruction

The SAR wavefront imaging [born], [mor], [s92], [s99] method is a Fourier-based approximation-free algorithm that provides high-resolution and accurate coherent target information that is useful for advanced SAR information post-processing. The basic principle for image formation is a Fourier (Doppler) processing of the SAR phase history signal that relates the SAR signal 2D spectrum $S(\omega, k_u)$ directly to the the target function 2D spectrum $F(k_x, k_y)$. The wavefront algorithm was originally introduced for linear SAR systems. The algorithm yields SAR images that are superior to the images that are formed via the backprojection algorithm; the computational cost of the wavefront algorithm is also significantly less than that of the backprojection method.

In recent years, national security and safety issues created the need for developing FOliage PENetrating (FOPEN) SAR systems that utilize Ultra WideBand (UWB) UHF/VHF radars for detection of concealed targets [s94], [s94], [s99], [s01b]. These *wide-beamwidth* FOPEN systems could be viewed as *non-linear* SAR due to large motion errors across their long synthetic apertures. The SAR wavefront digital signal processing issues associated with these FOPEN reconnaissance SAR systems brought new complexities and misunderstandings for those familiar with the traditional SAR systems. This is mainly due to the wide-bandwidth of these SAR systems. In fact, various FOPEN SAR processors which are found in the literature suffer from various misunderstandings, misconceptions, and incorrect implementation of the SAR wavefront reconstruction [s01a], [s01b].

Many who failed in implementing the wavefront-based method suggested the use of the simple correlation-based backprojection method despite its high computational cost. These

authors also claimed that the Fourier-based wavefront algorithm cannot handle non-uniform motion errors over long synthetic apertures, i.e., a non-linear SAR scenario. However, the tools that are provided by SAR wavefront-based signal processing and, in particular, the mapping of the aspect angle domain into the Doppler domain via the Fourier properties of AM-PM signals, enables a user to apply this algorithm in non-linear SAR systems.

Note that due to inaccuracies in the navigational and GPS data, range-gate slip, imperfections in the radar radiation pattern, etc., no matter which algorithm is used (e.g., back-projection, wavefront, etc.), the user has to perform further motion compensation and/or 2D focusing on the formed image using, e.g., in-scene targets, entropy maximization methods, etc., particularly for high-resolution (e.g., X or Ku band) SAR systems.

4.2 Subaperture-Based Recursive Video Image Formation

The basic principle behind the wavefront-based non-linear SAR imaging is to form lower-resolution images of the target area from subsets of the synthetic aperture or synthetic *subapertures* over which the flight path can be approximated by a linear path plus some *known* residual motion errors (i.e., the difference between the actual non-linear path within a subaperture and the linear path that is used to approximate it); this is shown in Figure 6.

The *slope* of the line that is used to approximate each subaperture could vary. The sizes of the subapertures is determined based on the non-linear path of the radar-carrying aircraft and/or the rate at which the video SAR is to be updated/refreshed. The scenario that is shown in Figure 6 involves subapertures with varying length; this is an option. The user may use subapertures that are equal in length.

Note that a subaperture is typically comprised of about a hundred or less PRIs. In this case, with a PRF of, e.g., 1 kHz and an aircraft speed of 150 m/sec, it is physically impossible for the aircraft to have rapid jumps within a subaperture slow-time interval. Thus, a *linear* approximation for the flight-path within a subaperture is a valid assumption. Again we should point out that the user should also apply appropriate residual motion error (i.e., the difference between the actual non-linear path within a subaperture and the linear path that is used to approximate it) compensation; this is a straightforward operation.

It should also be noted that a typical video that is used for human inspection has a refresh rate of 10-30 Hz. With a PRF of 1 kHz and a refresh rate of 20 Hz, the number of PRIs used for each subaperture processing is about 50; this is within the desired subaperture size that we mentioned earlier that can be approximated by a linear flight path.

The key for the success of the algorithm is that the *approximation-free* wavefront-based subaperture imaging algorithm preserves the coherent information among the lower-resolution images. Thus, the user could coherently add the lower-resolution images to form the desired high-resolution SAR image. Figure 6 shows the parameters that are associated with the low-resolution image formation with Subaperture 4. For this subaperture, the squint angle and squint range with respect to the scene center are (θ_{c4}, R_{c4}). The conventional reconstruction is typically formed in the corresponding range and azimuth domain of $(x_{\theta_{c4}}, y_{\theta_{c4}})$.

However, since we are interested in forming the image in the (desired) coordinate system (x, y), a 2D interpolation is used to translate the 2D spectrum of the SAR signal for

Subaperture 4, call it $S_4(\omega, k_{u\theta_{c4}})$, into a *segment* of the target function 2D spectrum in the desired coordinate system spatial frequency domain, that is, (k_x, k_y); the variable $k_{u\theta_{c4}}$ is the Doppler domain for the linear path that is used to approximate Subaperture 4.

We identify the segment of the target 2D spectrum that is formed by Subaperture 4 data via $FS_4(k_x, k_y)$. The inverse 2D Fourier transform of this 2D spectrum yields a low-resolution image; however, that low-resolution image on its own is not sufficient for visualization. Suppose the user wishes to integrate the data within K_s subapertures to form a single frame of the *high-resolution* video SAR image. Thus, after acquiring the SAR data for the subaperture number N_s, the user forms the 2D spectrum of the new frame via

$$F_{N_s}(k_x, k_y) = \sum_{n=N_s-K_s+1}^{N_s} FS_n(k_x, k_y). \tag{18}$$

The 2D spectrum that is constructed for the previous frame is

$$F_{N_s-1}(k_x, k_y) = \sum_{n=N_s-K_s}^{N_s-1} FS_n(k_x, k_y). \tag{19}$$

Thus, we can use the following *recursive* or *updating* equation to form the 2D spectrum of the N_s-th frame:

$$F_{N_s}(k_x, k_y) = F_{N_s-1}(k_x, k_y) + FS_{N_s}(kx, k_y) - FS_{N_s-K_s}(k_x, k_y). \tag{20}$$

To form the spatial domain image (updated frame), the 2D inverse Fourier transform of the above 2D spectrum is obtained. The main challenge that we face for the wavefront-based video non-linear SAR image formation is to incorporate all the necessary phase functions for each subaperture data that would accurately *calibrate* the subaperture-based target spectra. We have successfully demonstrated this concept with a wide-bandwidth (12,000 PRIs per pixel) FOPEN SAR database [s01a], [s01b]. The algorithm has also been used to form images for a 4 GHz X-band spotlight SAR system with an integration angle of over 30 degrees (around 80,000 PRIs) with noticeably non-linear flight path. (The results have not been published, and are available at the Air Force's Wright Laboratory.) That approach forms the basis for the formulation of the general subaperture-based video image formation in non-linear SAR.

4.3 Wavefront Reconstruction Program and Parallelization

The original Matlab program is wave_xy4_sim.m. It was parallelized using MatlabMPI and a GUI was created for it by using *guide*. The parallel version is named wave.m and GUI files start with wave_function.m. Figure 7 shows the wavefront reconstruction output image.

The main structure of the code is as follows:

```
... (Seq. code)
for (ia =1:n_sub_aper)
    ... (Outer for loop code)
    for (is=1:ns)
    ... (Inner for loop code)
```

```
    end
end
... (Seq. code)
```

Several different approaches can be taken in parallelizing this code. But because the program must be able to run on any data input parameters and on any number of processors, the outer *for* loop parallelization is adopted. The user first specifies how many processors are available for parallel computation. Depending on the input parameters, the number of iterations in the outer *for* loop and the inner *for* loop will differ. The number of iterations in the outer *for* loop is equal to the input parameter "n_sub_aper".

To describe the parallelization scheme, we use a simple example.

If "n_sub_aper=6" and "procs=4" (i.e. there are six iterations in the outer *for* loop and four processors available), the 1st session will execute "ia=1", 2nd session will execute "ia=2", the 3rd session will execute "ia=3", the 4th session will execute "ia=4", the 1st processor will execute "ia=5" after it has finished the "ia=1" iteration, and 2nd processor will execute "ia=6" after it has finished the "ia=2" iteration.

This way, the program will execute concurrently until all the iterations in the outer *for* loop has been executed by the processors. For each iteration in the outer *for* loop, a data file is created that contains the results of each iteration.

Simulation Results

Table 3 shows the runtime of the parallelized wavefront reconstruction program. Every set of parameters is run 5 times and the average runtime is shown. From this table, we can find that Hellfire is generally two times faster than Stingray. Besides that, it behaves similarly to Stingray. When the number of nodes is increased, the runtime is decreased almost proportionally.

Figure 8 shows the speedup as a function of the number of nodes for the wavefront

Table 3: Performance Comparison of the Wavefront Reconstruction Program

Number of nodes	1	2	4	8	8	8	16	32
range(m)	1000	1000	1000	1000	1000	100	1000	1000
azimuth(m)	1000	1000	1000	1000	100	1000	1000	1000
Error in offset(m)	0	0	0	0	0	0	0	0
Number of sub-sub-apertures	1	1	1	1	1	1	1	1
Number of sub-apertures	18	18	18	18	18	18	18	18
Runtime(sec) on Stingray	365.2	173.9	110.7	71.6	53.3	39.3	66.0	N/A
Runtime(sec) on Hellfire	172.2	104.1	59.6	41.0	25.2	18.0	29.0	15.0
Ratio (Hellfire/Stingray)	47.2%	59.9%	53.8%	57.3%	47.3%	45.8%	43.9%	N/A

reconstruction program. Due to the parallelizing structure of this program, it can be observed that the speedup mainly occurs when the number of nodes is the factor of 18. Because the runtime of the serial version is only about 3 minutes, the results seen here are quite volatile.

This is not the case of the other two programs, both of which need multiple hours to run; therefore the results are more steady and valuable.

5 Multi-Channel SAR-MTI Processing

5.1 Overview

The Department of Defense has been developing prototype multi-channel airborne radar systems to improve its capability for intelligence gathering, surveillance and reconnaissance. These airborne array radar systems are intended to collect rich databases that could be exploited for various tasks such as moving target detection and tracking, target imaging and recognition, etc. The multi-channel airborne radar systems provide a large volume of multidimensional information regarding the imaging scene that is being interrogated. For these systems, the task of a radar signal processor is to develop information processing algorithms that fully exploit the measured large-volume multi-channel airborne radar databases. Some of the issues include interpreting and/or coherently combining the airborne array data via imaging algorithms of Synthetic Aperture Radar (SAR), calibrating the data in various channels to detect subtle information that are critical for Airborne/Ground Moving Target Indicator (AMTI/GMTI) problems, etc.

Multi-Channel Airborne Radar Measurement (MCARM) system [slo] developed by the Air Force Research Laboratory at Rome, New York, is one of the data acquisition programs that was established in support of the above-mentioned objectives. Under the MCARM program, multichannel clutter data were collected using an L-band active aperture and multiple IF receivers. The data were collected at a variety of pulse repetition frequencies (PRF) and over various terrains including mountains, rural, urban, and land/sea interface. The MCARM data were obtained from a multi-channel sub-aperture architecture and a low sidelobe sum and difference analog beamformer. The multi-channel architecture is a sub-aperture configuration with 22 degrees of freedom (receivers).

The MCARM data were primarily collected for the exploration of adaptive array processing algorithms such as Space Time Adaptive Processing (STAP). In a recent paper [s02], we introduced methods to interpret and process the MCARM data via the SAR-based imaging and MTI algorithms. Such SAR-based signal processing algorithms had not been attempted with the MCARM data, since this airborne radar system possessed a relatively small bandwidth (0.8 MHz) and synthetic aperture (less than 8 m at its medium PRF of 2 kHz). These result in a relatively poor range-dependent resolution, e.g., 150 by 150 m at the range of 8 km. Note that certain conventional SAR systems possess even worse resolution. Moreover, the signal measured via an individual receiver (with an aperture of a few centimeters) did not have sufficient power to yield an image with a good fidelity.

The fundamental concept that we exploited in [s02] was to interpret and process the MCARM data within the governing principles for an along-track monopulse SAR system [s97]. The along-track monopulse SAR imaging system utilizes two radars for its data collection. One radar is used as a transmitter as well as a monostatic receiver. The other radar is used only as a bistatic receiver. We have developed a two-dimensional adaptive filtering method, called Signal Subspace Processing (SSP) [s99, Ch 8], to blindly calibrate the two channels of an along track monopulse SAR system. In this case, the two monostatic and

bistatic databases of the along track monopulse SAR system yield two coherently identical SAR images of the stationary targets in the scene. While the stationary targets appear the same in the monostatic and bistatic SAR images, however, the same is not true for moving targets. Thus, the difference of the SSP-calibrated monostatic and bistatic data yields a statistic that is suitable for AMTI/GMTI.

5.2 MCARM SAR-MTI Processing

The MCARM system used in this study is shown in Figure 9. In the transmit mode, all the sub-apertures of the system are used in a phased array configuration to radiate the target scene. The data were collected using a wide-beamwidth and narrow-beamwidth radiation patterns (via appropriate phasing of the transmitting sub-apertures.

In the receive-mode, 22 sub-apertures were used to record the echoed signals. In Figure 9, these are identified as Modules 2-8 and 10-24. For simplicity, we refer to these as Receiver Elements 1-22. Our study (calibration results) indicates that the transmit-mode phase (time) delays were not turned off during the individual receptions of the 22 elements.

For the n-th receiving element, the continuous domain measured signal can be identified as $s_n(t,u)$, where t represents the fast-time domain (in seconds) and u is the synthetic aperture domain (in meters). Let the (Doppler) Fourier transform of the received signal with respect to the slow-time be

$$S_n(t,k_u) = F_{(u)}[s_n(t,u)]. \tag{21}$$

Then, the target reconstruction is achieved via the following mapping:

$$f(x,y) = S_n(t,k_u), \tag{22}$$

where

$$x = r\cos\theta \quad and \quad y = r\sin\theta \tag{23}$$

and

$$r = \frac{ct}{2} \quad and \quad \theta = \arcsin\Big(\frac{k_u}{2k_c}\Big). \tag{24}$$

In the above, c is the wave propagation speed, and k_c is the wavenumber at the carrier radar frequency.

Note that in theory the above target function can be formed using the received signal of any of the 22 elements. However, in practice due to noise and other sources of errors, a one-element imaging does not yield a high fidelity image. Thus, our first task is to combine/add the data of the 22 elements. However, there is a practical impasse in doing so; this is described and treated next.

We call the signal that is recorded by the first element, i.e., $s_1(t,u)$, the reference signal. For a stationary scene and when the elements possess a common radiation pattern, the signal that is recorded by the n-th element can be related to the reference signal via

$$s_n(t,u) = a_n s_1(t, u+u_n). \tag{25}$$

a_n is a complex number that represents a difference in gain and phase of the two signals; one source of this gain/phase is the relative physical distance of the n-th element from the

first element in the slant-range domain. u_n is the relative shift of the data in the slow-time domain; this shift is primary due to the relative physical distance of the n-th element from the first element in the along-track (azimuth) domain. Both (a_n, u_n) are also dependent of the internal circuitry of the two elements.

The above is a relatively simplistic way to relate the recorded signals, which assumes that the signals that are recorded by the two elements are related by a global gain/phase and slow-time delay (both of which are unknown). We will treat this problem via a more complicated model later. However, to get started, we estimate these global parameters (a_n, u_n) via constructing the two-dimensional (2D) cross-correlation of $s_n(t, u)$ and the reference signal $s_1(t, u)$.

We identify the globally-calibrated signal for the n-th receiver channel via

$$\hat{s}_n(t, u) = \frac{1}{\hat{a}_n} s_n(t, u - \hat{u}_n). \tag{26}$$

Consider the calibrated data from any two of the 22 receiver elements, e.g., n=1 and 2. Let $\hat{S}_1(t, k_u)$ and $\hat{S}_2(t, k_u)$ be the slow-time (Doppler) Fourier transforms of $\hat{s}_1(t, u)$ and $\hat{s}_2(t, u)$, respectively. Provided that all the 22 channels were calibrated, the user could use the following statistic for MTI/GMTI:

$$\hat{S}_d(t, k_u) = \hat{S}_2(t, k_u) - \hat{S}_1(t, k_u). \tag{27}$$

Let $f_2(x, y)$ and $f_1(x, y)$ be the SAR images that are formed from $\hat{S}_2(t, k_u)$ and $\hat{S}_1(t, k_u)$, respectively. In this case, an equivalent MTI/GMTI information can be constructed in the target scene reconstruction domain via

$$f_d(x, y) = f_2(x, y) - f_1(x, y). \tag{28}$$

Note that $f_d(x, y)$ is the SAR image that can be formed from $\hat{S}_d(t, k_u)$.

Due to various sources of errors (variations of the elements' radiation patterns in space, etc.), the global calibration of the 22 channels of MCARM would not be sufficient to guarantee the success of the above MTI processing. In fact, in practice the MTI/GMTI signatures are relatively weak, and can be dominated by even small miscalibrations.

To model these miscalibrations, we consider the two synthesized along-track channels of MCARM in the (t, k_u) domain or, equivalently, spatial (x, y) domain. The calibration errors result in a spatially-varying point spread function (PSF) or Image Point Response (IPR) [s99, Ch 8]. However, within a small sub-patch of the target domain (e.g., a few hundred meters in both range and azimuth), the miscalibration PSF could be assumed to be spatially-invariant. For example, if $\hat{S}_{2m}(t, k_u)$ and $\hat{S}_{1m}(t, k_u)$ are the data for the m-th sub-patch that are centered at $t = t_m$ and $k_u = k_{um}$, we can relate these two signals (when there is no moving target) via the following 2D convolution:

$$\hat{S}_{2m}(t, k_u) = \hat{S}_{1m}(t, k_u) \otimes h_m(t, k_u), \tag{29}$$

where the 2D (complex) filter (PSF) function $h_m(t, k_u)$ is unknown. Note that the filter function may also be represented as a spatially-varying PSF with respect to the center of the sub-patch via

$$h_m(t, k_u) = h(t, k_u; t_m, k_{um}). \tag{30}$$

A procedure to estimate the unknown filter function $h_m(t, k_u)$ via 2D adaptive filtering within a sub-patch is described in [s99, Ch 8]; this procedure is referred to as Signal Subspace Processing (SSP). Let $\hat{h}_m(t, k_u)$ be the resultant estimated filter. Then, the MTI/GMTI statistic for the m-th sub-patch is formed via:

$$\hat{S}_{dm}(t, k_u) = \hat{S}_{2m}(t, k_u) - \hat{S}_{1m}(t, k_u) \otimes \hat{h}_m(t, k_u). \tag{31}$$

We refer to the above as the Local Signal Subspace Difference (LSSD) image.

Our study of the along-track SAR data for MTI or change detection for various radar bands and SAR platforms has indicated that at times the Localized SSP algorithm not only removes the stationary targets but also performs a partial calibration with respect to a moving target at a sub-patch. This results in a weaker signature of the moving target in the LSSD image. To counter this problem, we hypothesized that since a radar system is a physical entity, the coefficients of the miscalibration filter/PSF $h_m(t, k_u)$ should not exhibit rapid fluctuations from one patch to its neighboring sub-patches for a stationary scene; a rapid change in a coefficient of the filter is most likely caused by adaptation of the SSP method to the signature of a moving target in that sub-patch.

Thus, the estimated filters that exhibit rapid fluctuations are not only unreliable but also are likely to be the ones that are adapting to moving target signatures and weakening their presence in the LSSD image. A simple remedy for this is to fit a low order 2D polynomial of the sub-patch location, i.e., (t_m, k_{um}) in the (t, k_u) domain to each estimated filter coefficient. The resultant smooth spatially-varying filter function can then be applied to Channel 1 signal $\hat{S}_1(t, k_u)$; the outcome is then subtracted form Channel 2 signal $\hat{S}_2(t, k_u)$ to form what we refer to as the Global Signal Subspace Difference (GSSD) image.

5.3 Synthesis of Along-Track SAR Channels

In the previous section, we presented the basic principles to interpret the data from a single receiver element of the MCARM system as a SAR signal. We then identified the data from two of the elements as the signal from an along-track SAR system, and an adaptive method for 2D adaptive calibration of such a SAR system was presented for MTI/GMTI purposes.

We now examine one method for converting the data from all the 22 receiver elements of the MCARM system into a dual channel along-track SAR database.

Our approach is based on using a pair of elements of the MCARM as the two channels of an along-track monopulse SAR system. We perform SSP on the resultant two channels. This step is then repeated using other pairs (231 cases); the resultant 231 SSP differences are added up to construct the information that is used for MTI. For this method, the coherent information in the individual elements are exploited in these 231 parings. Furthermore, since one anticipates more clutter and noise suppression in averaging the 231 SSP differences, this is the most promising approach for generating the MTI statistic.

In fact, when calibrating one element versus another element, a speckle pattern (noise) is generated due to subtle miscalibration at certain range and Doppler bins; however, when another pair is processed, the speckle noise miscalibraion at the same range-Doppler bin would most likely be different. This is known as speckle averaging effect in optics literature

[goo].

This method carries the highest computational burden. Yet, since the individual pairings are based on independent SSP operations that do not require the usage a significant amount of memory, the algorithm can be easily implemented on an on-board distributed memory multi-processor system.

5.4 Multi-Channel SAR-MTI Program and Parallelization

The original Matlab program is step2.m. It was parallelized using MatlabMPI. The parallel version is named step2_para.m. Figure 10 shows the Multi-Channel SAR-MTI output image.

There are two *for* loops in step2.m for which parallelization is required. The structure of those loops in the sequential program is as follows:

```
for (im = 1: 21)
    for (jm = im+1: 22)
        ...(Computation algorithm)
    end
end
```

Compared with the backprojection program, it is a little more difficult to convert the two *for* loops into a single *for* loop. Therefore, we retain the two *for* loops structure and add a counter in the parallelized program. The new structure is as follows:

```
ic=0;
for (im = 1: 21)
    for (jm = im+1:22)
        ic=ic+1;
        rank_run = mod(ic-1,procs);
        if (rank_run == my_rank)
        ...(Computation algorithm)
        end
    end
end
```

Through this process, the original 231 loops are evenly distributed to every available processor. Unlike the previous program, it is not predetermined that a specific processor should perform which loop. Instead, the assignment of each loop is determined inside the loop. Although this structure wastes a little time, it ensures that only minor changes need to be made to the sequential program. This greatly reduces the burden of the programmer and makes the parallelized program more readable and understandable. Like the backprojection program, at the end of each loop, the slave processors send their computed data to the master. The master then combines these data and outputs the result.

Simulation Results

Figure 11 shows the speedup for Multi-Channel SAR-MTI Program. This is an "embarrassingly parallel" program and theoretically the speedup can reach infinity. However,

the effect of communication overhead will kick in when the number of nodes is large enough. As we can see from Figure 11, the speedup gradually deviates from the theoretical boundary when the number of node goes up.

To assure the transplantation of the parallelized algorithms from shared-memory system

Table 4: Slightly Different Results on Different Machines

Program	Average Difference	Maximum Difference
Backprojection	4.6923×10^{-15}	2.2893×10^{-13}
Wavefront Reconstruction	3.1670×10^{-15}	3.6238×10^{-13}
Multi-Channel SAR-MTI	7.3025×10^{-19}	2.4286×10^{-17}

Stingray to distributed-memory system Hellfire is correct, we also compared the results from both machines. Table 4 shows the difference of the results of both programs. This difference is mainly due to the fact that the MATLAB on two machines utilizes different Basic Linear Algebra Subroutines (BLAS) libraries. BLAS libraries are used to speed matrix multiplication and LAPACK functions. MATLAB usually includes multiple versions of these BLAS libraries which are optimized for different processors. At startup, MATLAB automatically detects the type of the processor and select the most appropriate BLAS library. One can actually set the environment variable LAPACK_VERBOSITY to let MATLAB display the version of BLAS library being used.

E.g.:

>> setenv LAPACK_VERBOSITY 1; *% On Stingray*

>> export LAPACK_VERBOSITY=1; *% On Hellfire*

From the startup message of MATLAB, we can know that the BLAS library used on Hellfire is atlas_P4.so, and the BLAS library used on Stingray is atlas_R12000.so.

6 Conclusions

The parallelization of the nonlinear SAR algorithms and multi-channel SAR-MTI processing was successfully implemented on two different machines by using MatlabMPI. Our study has shown that the parallelized programs share similar performance in spite of different running environment. Communication overhead is always a big obstacle to efficient parallelization. Thanks to careful arrangements, communication was utilized to a minimum in all three programs and was not a big issue. Besides communication, f (the fraction of the program which cannot be parallelized) is the most important limiting factor in parallelization. This suggests that reducing f should be the focus of the future study. We are currently in the process of implementing these programs by using pMatlab - the next generation of MatlabMPI released by MIT Lincoln Laboratory. The concept of distributed matrix introduced by pMatlab holds promise for parallelizing most part of a specific program, if not the entire program.

7 References

[bar] B. Barber, "Theory of digital imaging from orbital synthetic aperture radar," *International Journal of Remote Sensing*, vol. 6, no. 7, pp. 1009-1057, July 1985.

[born] M. Born and E. Wolf, *Principles of Optics*, 6th edition, New York: Pergamon Press, 1983.

[gab] D. Gabor, "A new microscope principle," *Nature*, vol. 161, p. 777, 1948.

[goo] J. W. Goodman, *Statistical Optics,* New York: Wiley, 1985.

[mor] P. M. Morse and H. Feshbach, *Methods of Theoretical Physics*, New York: McGraw-Hill, Parts 1 and 2, 1953.

[slo] D. Sloper et al., MCARM Final Rept., Rome Lab Tech. Rept., RL-TR-96-49, April 1996, also sees http://sunrise.deepthought.rl.af.mil.

[s92] M. Soumekh, "A system model and inversion for synthetic aperture radar imaging," *IEEE Transactions on Image Processing*, vol. IP-1, no. 1, pp. 64-76, January 1992.

[s94] M. Soumekh, *Fourier Array Imaging*, Englewood Cliffs, NJ: Prentice Hall, 1994.

[s95] M. Soumekh, "Reconnaissance with ultra wideband UHF synthetic aperture radar," *IEEE Signal Processing Magazine*, vol. 12, no. 4, pp. 21-40, July 1995.

[s97] M. Soumekh, "Moving Target Detection in Foliage Using Along Track Monopulse Synthetic Aperture Radar Imaging," *IEEE Transactions on Image Processing*, vol.6, no. 8, pp. 1148-1163, August 1997.

[s99] M. Soumekh, *Synthetic Aperture Radar Signal Processing,* New York: Wiley, 1999.

[s01a] M. Soumekh, "Wavefront-Based Synthetic Aperture Radar Signal Processing," *Frequenz*, pp. 99-113, March/April 2001 (special issue on *Advanced SAR Techniques*).

[s01b] M. Soumekh, D. Nobels, M. Wicks and G. Genello, "Signal Processing of Wide-Bandwidth and Wide-Beamwidth P-3 SAR Data," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 4, pp. 1122-1141, October 2001.

[s02] M. Soumekh and B. Himed, "SAR-MTI Processing of Multi-Channel Airborne Radar Measurement (MCARM) Data," *Proc. IEEE Radar Conference*, Long Beach, May 2002.

8 Figures

1. Non-linear SAR System Geometry.
2. Block Diagram for Linear and Non-Linear SAR Image Formation Using Backprojection Algorithm.
3. Backprojection Output Image.
4. Theoretical Maximum Speedups for Different Values of f.
5. Speedup vs Number of Nodes (Backprojection Program).
6. Subaperture Processing for Non-Linear SAR.
7. Wavefront Reconstruction Output Image.
8. Speedup vs Number of Nodes (Wavefront Reconstruction Program).
9. MCARM System.
10. Multi-Channel SAR-MTI Output Image.
11. Speedup vs Number of Nodes (Multi-Channel SAR-MTI Program).

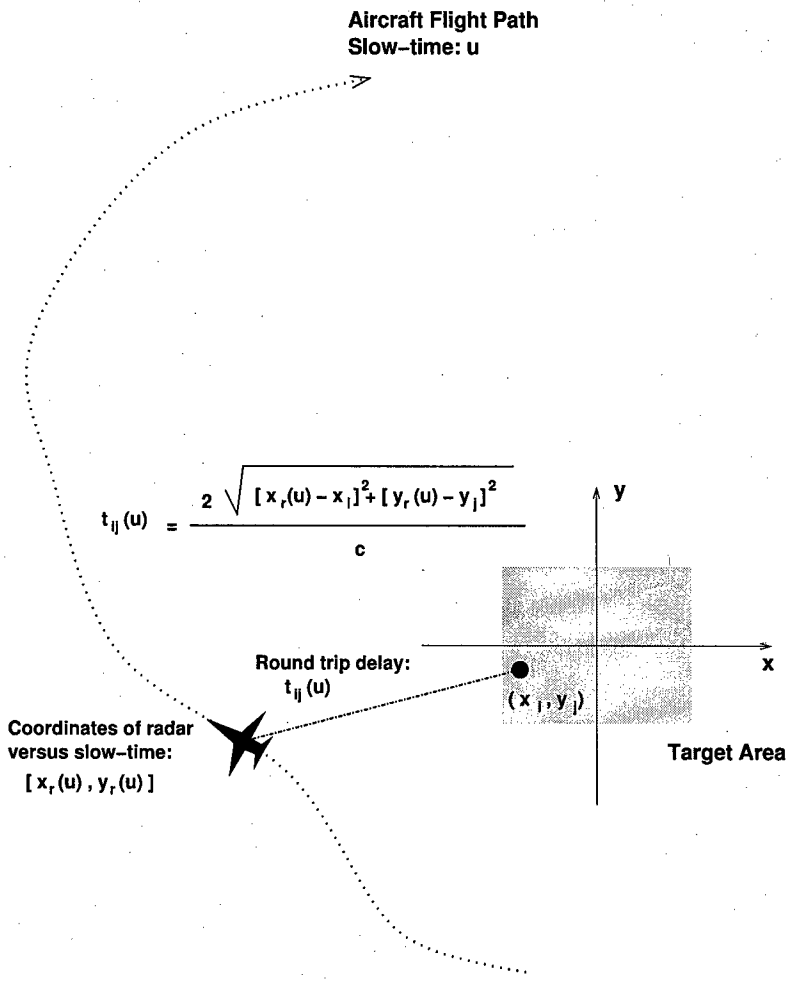

Figure 1. Non-linear SAR System Geometry

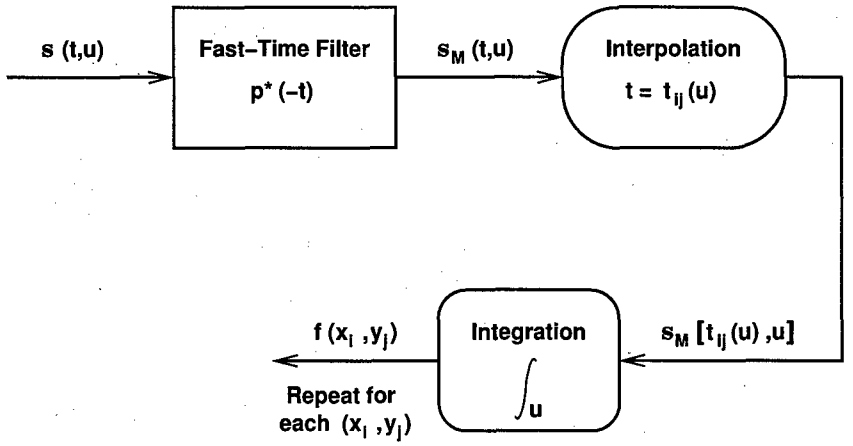

Figure 2. Block Diagram for Linear and Non-Linear SAR Image Formation Using Backprojection Algorithm

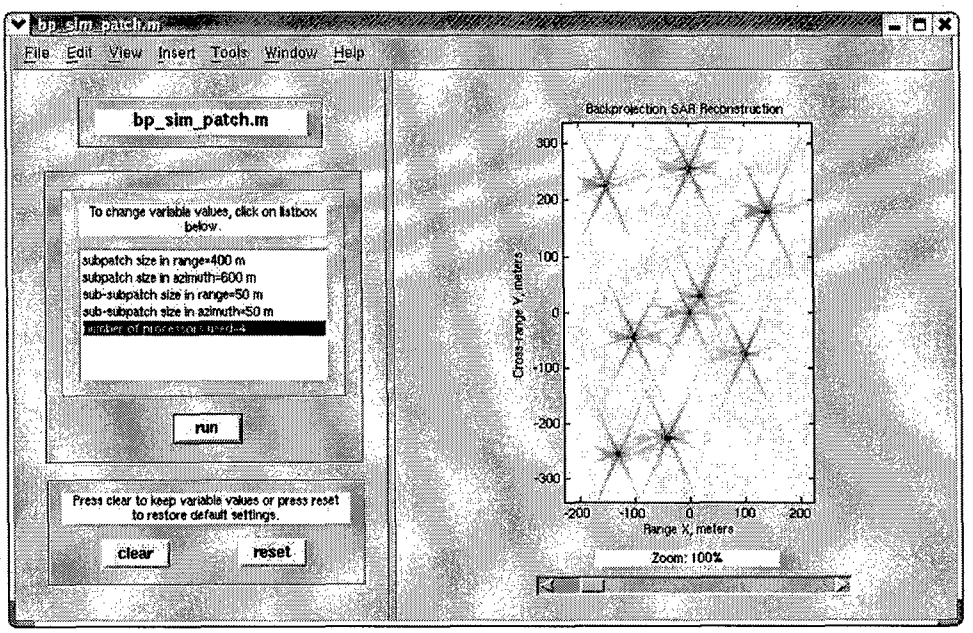

Figure 3. Backprojection Output Image

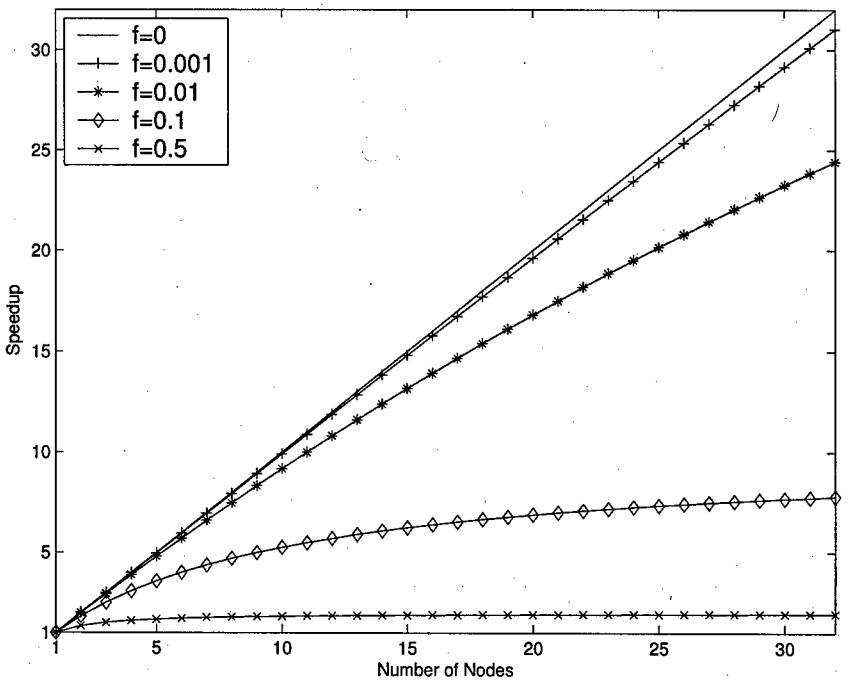

Figure 4. Theoretical Maximum Speedups for Different Values of f

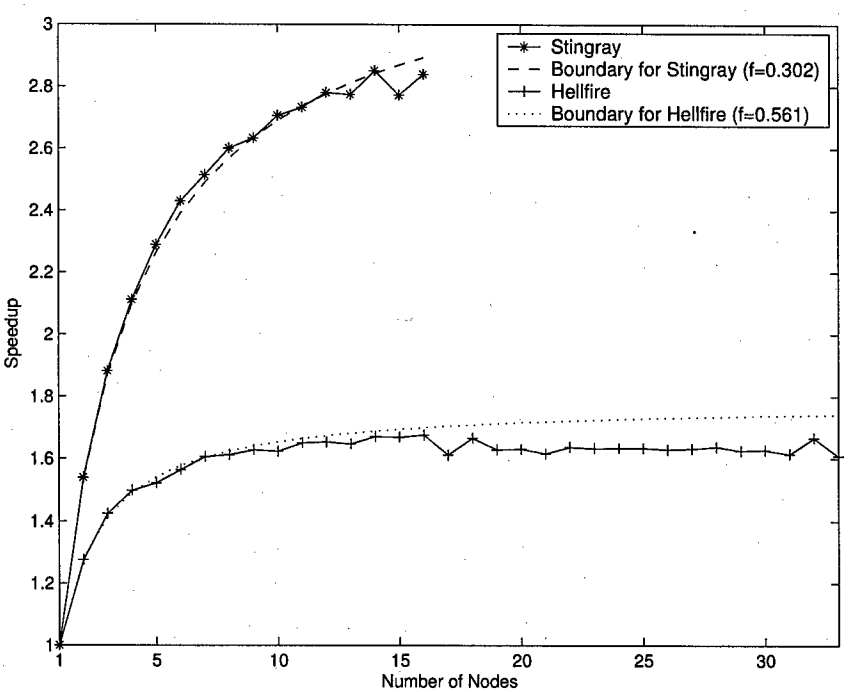

Figure 5. Speedup vs Number of Nodes (Backprojection Program)

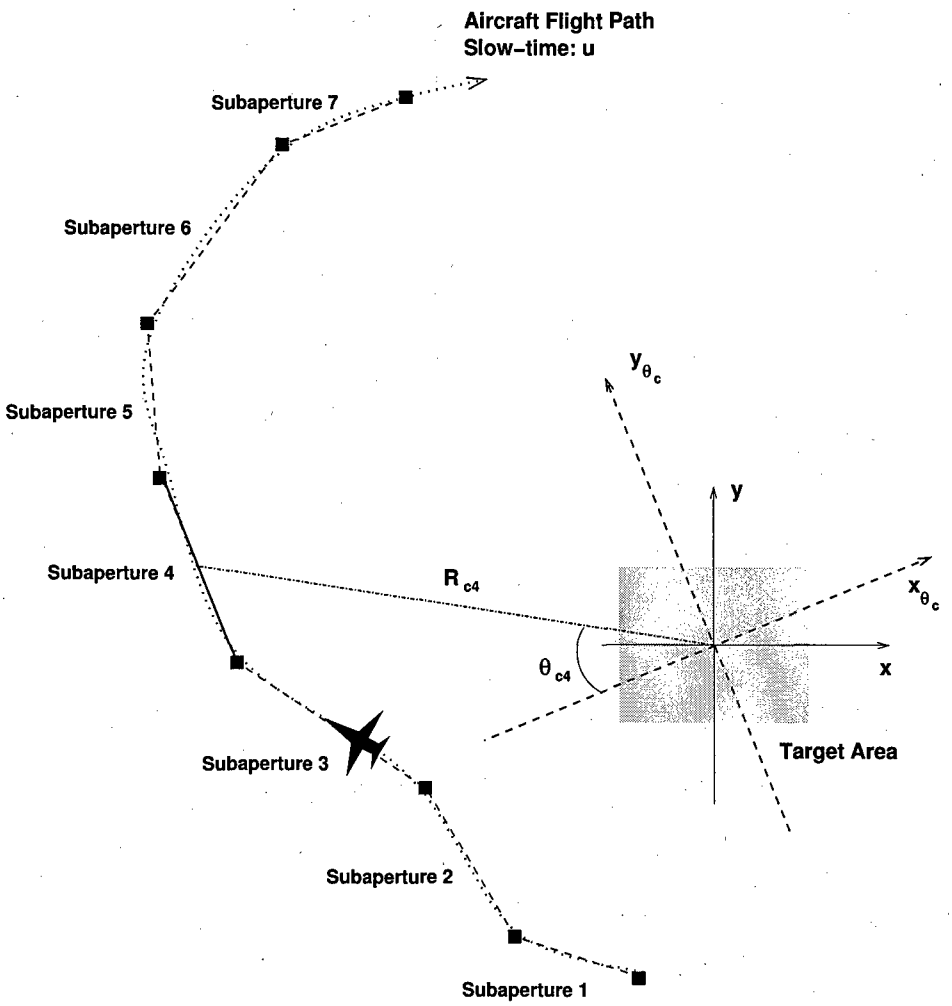

Figure 6. Subaperture Processing for Non-Linear SAR

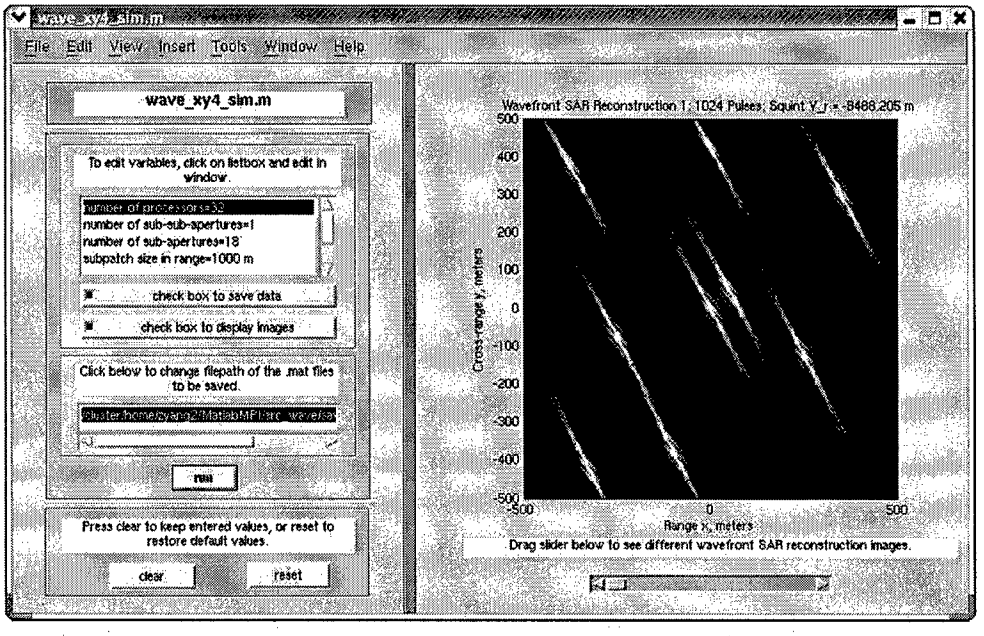

Figure 7. Wavefront Reconstruction Output Image

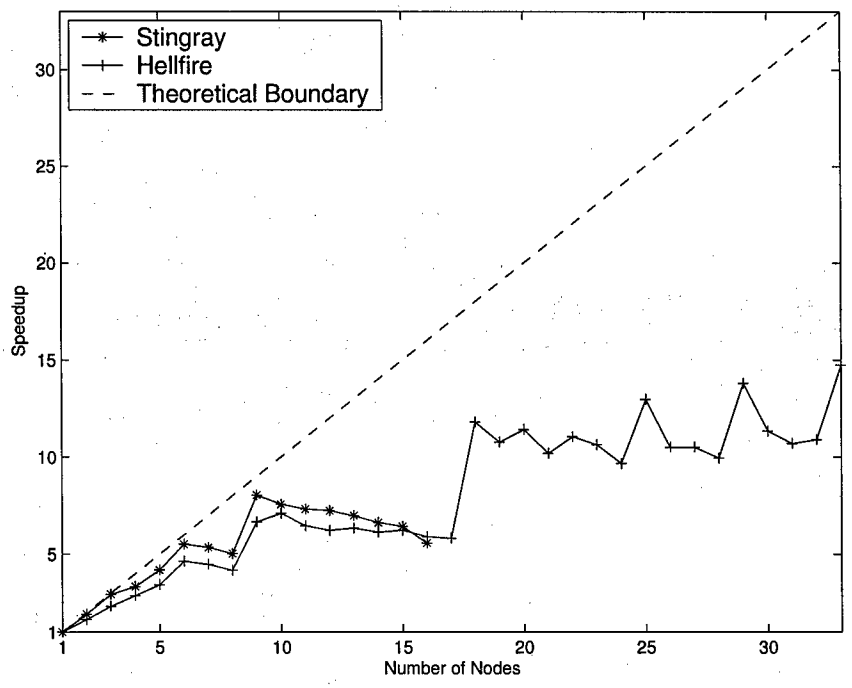

Figure 8. Speedup vs Number of Nodes (Wavefront Reconstruction Program)

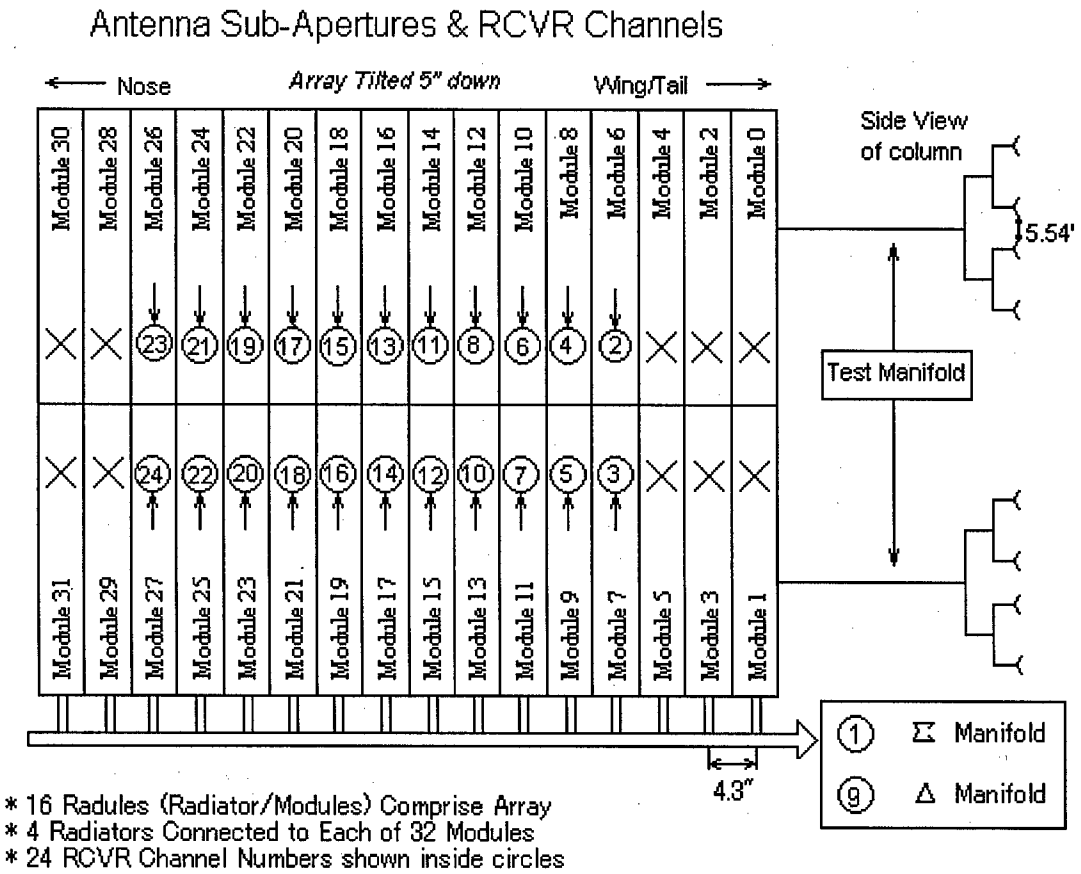

Figure 9. MCARM System

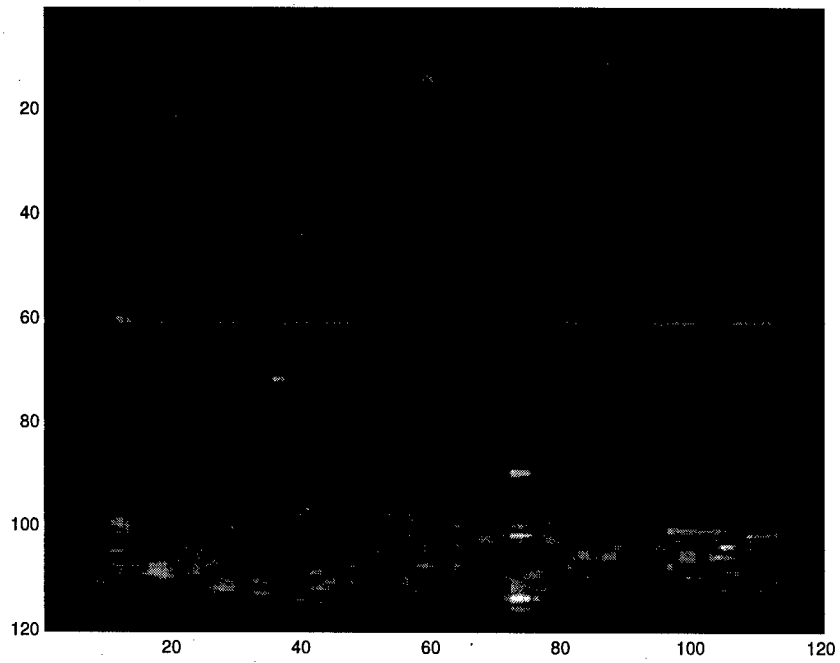

Figure 10. Multi-Channel SAR-MTI Output Image

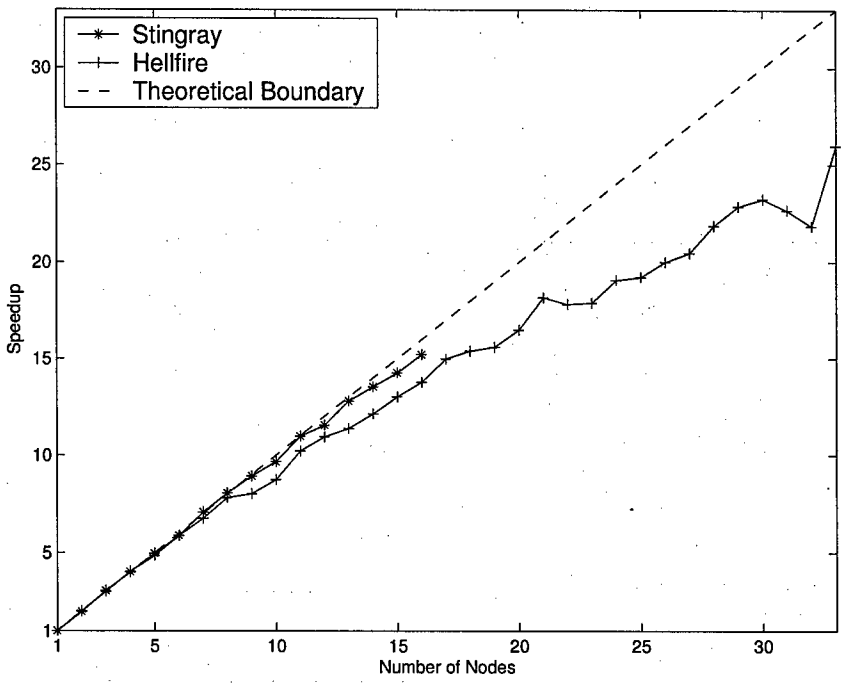

Figure 11. Speedup vs Number of Nodes (Multi-Channel SAR-MTI Program)